In the Claims

Please replace all prior versions, and listings, of claims in the application with the following listing of claims, including marked-up claims with insertions indicated by underlining and deletions indicated by strikeouts and/or double bracketing:

1. (Currently Amended) A method of compiling a computer program from a sequence of computer instructions ~~including a plurality of first, set branch, instructions which each identify a target address for a branch and a plurality of associated second, effect branch instructions which each implement a branch to a target address~~, the method comprising:

reading, in blocks, said computer instructions including a plurality of first, set branch, instructions which each identify a target address for a branch and a plurality of associated second, effect branch instructions which each implement a branch to a target address ~~in blocks~~;

defining a set of target registers associated with each block for holding target addresses for the set branch instructions in that block;

defining as a live range of blocks a set of blocks for which a target address of a particular set branch instruction is in a live state; ~~and~~

using said set of target registers and said live range to ensure that target registers holding target addresses in a live state are not available for other uses;

for each set branch instruction, allocating the set branch instruction to a respective initial node in a dominator tree, said initial node being the node which contains the effect branch instruction corresponding to the set branch instruction;

determining an ancestor node in the dominator tree to which to migrate one or more of the branch instructions based on the live range of blocks so that target registers holding target addresses in a live state are not overwritten when the computer instructions are executed; and

migrating one or more said branch instruction to the ancestor node.

2. (Cancelled)

3. (Original) A method according to claim 2, wherein, during said step of migrating said at least one set branch instruction, the live range of blocks is incrementally updated.

4. (Original) A method according to claim 3, wherein during said step of migrating said at least one set branch instruction, the set of target registers holding target addresses in a live state is simultaneously incrementally updated.

5. (Original) A method according to claim 1, wherein the union of said set of target registers and said live range is taken to define target registers holding target addresses in a live state.

6. (Currently Amended) A method of operating a computer system to compile a computer program from a sequence of computer instructions ~~including a plurality of first, set branch instructions which each identify a target address for a branch and a plurality of second, effect branch instructions which each implement a branch to the target address specified in the associated set branch instruction~~, the method comprising:

executing a dominator tree constructor function in the computer system to read, in blocks, said computer instructions ~~in blocks~~ including a plurality of first, set branch instructions which each identify a target address for a branch and a plurality of second, effect branch instructions which each implement a branch to the target address specified in the associated set branch instruction and to define a set of target registers associated with each block for holding target addresses for the set branch instructions in that block;

executing a lifetime tracking algorithm to define as a live range of blocks a set of blocks for which a target address of a particular set branch instruction is in a live state, said lifetime tracking algorithm being operable to use said set of target registers and said live range to ensure that target registers holding target addresses in a live state are not available for other uses.

7. (Previously Presented) A method according to claim 6, which comprises the step of executing a migration function which migrates at least one set branch instruction to an ancestor node in the dominator tree.

8. (Previously Presented) A method according to claim 7, wherein said lifetime tracking algorithm is operable to define said live range of blocks on an incremental basis as the at least one set branch instruction is migrated.

9. (Currently Amended) A compiler for compiling a computer program from a sequence of computer instructions ~~including a plurality of first, set branch instructions which each identify a target address for a branch and a plurality of associated second, effect branch instructions which implement a branch to the target address specified in the associated set branch instruction~~, the compiler comprising:

a dominator tree constructor for reading, in blocks, said computer instructions ~~in blocks~~ including a plurality of first, set branch instructions which each identify a target address for a branch and a plurality of associated second, effect branch instructions which implement a branch to the target address specified in the associated set branch instruction and for allocating each set branch instruction to an initial node in a dominator tree, said initial node being located in the block which contains the corresponding effect branch instruction;

circuitry for defining a set of target registers associated with each block for holding target addresses for the set branch instructions in that block;

circuitry for executing a lifetime tracking algorithm which defines as a live range of blocks a set of blocks for which a target address of a particular set branch instruction is in a live state, and which is arranged to use said set of target registers and said live range to ensure that target registers holding target addresses in a live state are not available for other uses.

10. (Previously Presented) A compiler according to claim 9, which comprises a migration function for migrating a set branch instruction to one of said ancestor nodes in the dominator tree.

11. (Previously Presented) A compiler according to claim 10, which comprises a determiner for determining the effect of migrating said set branch instruction to each of a set of ancestor nodes in the dominator tree based on a performance cost parameter.